IP Fragmentation
Implementation scheme
Mon, June 26, 1992

Internet Protocol (IP) specifies support for fragmentation and reassembly of packets to form datagrams. The implementation for IP support in the local stations must include this feature. This note discusses how it is done.

Maximum datagram size

Theoretically an IP datagram, including the IP header plus message, can be 64K bytes in length. In practice, however, some implementations put a smaller upper limit for reasons of memory constraints and economy. According to Douglas Comer, a general expert in the field, timesharing systems typically choose values in the range of 4K–8K bytes. (See "Internetwork ing with TCP/IP Volume II" page 32.) With the token ring 4Mb network, the hardware frame length limit is about 4K bytes, so it is natural to consider adopting a local limit of 4K bytes for the maximum datagram size. The ethernet frame size limit is 1500 bytes, which is too low to consider for a datagram size limitation.

A limit of about 2004 bytes is detectible sending "multinet ping" requests from the Vax. (It may only be a ping client program limit.) The Vax limit on datagram replies to ping requests is at least 5000 bytes, the limit of the local station's current software. The Vax's UDP echo server limit seems to be 4144 bytes.

MTU size

Separate from the issue of datagram size limit is the choice of Maxi mum Transmission Unit, or MTU. This is the limit on fragment size transmitted on a physical network. With ethernet networks this limit is almost always 1500 bytes, the same as the hardware frame limit. In the Sun that uses the SBus token ring board, the MTU seems to be 2044 bytes for the size of the IP datagram. From RFC-1122, it is recommended that one choose an MTU for use with foreign net works (destination network id different from local) that is 576 bytes, a generally accep table limit in use throughout the world-wide Internet. From tests with some far-away IP nodes, however, it seems that 1448 byte datagrams work without further fragmentation being imposed.

Whatever the local decision is regarding maximum datagram size and MTU, it is necessary to support reassembly of IP packet fragments. So this support is needed for the local station IP support software.

Reassembly

The local station uses the IPARP table to hold IP addresses and hardware addresses and related port#s. The IP address is the key to this table. An entry in this table is given by a pseudo node# word, which is an index to a table entry and also identifies a port#; thus, it fulfills a function analogous to a socket

When an IP fragment is received, specified by the MF (more fragments) bit set in the IP header and/or a nonzero fragment offset value, something special must be done with it; otherwise, it simply remains in the DMA buffer written into by the chipset to await higher level processing. To support fragments, it is necessary to use a timeout in case not all fragments arrive that are to build a datagram. After each fragment is received, the timer should be reset. A timeout commonly in use is about 60 seconds. After a timeout, the fragments must be discarded. (If frag ment #0 has been received at that time, then an ICMP "time exceeded" error mes sage is sent to the source host. See RFC-1122.) The point is that fragments may have to wait a long time before they can be built into a datagram, so they can not remain in the 64K DMA buffer, which can wrap in a second or so under heavy traffic. Allocated memory can hold the fragment for later reassembly.

It is not convenient to build the datagram as the fragments are received, because the total size of the expected datagram is unknown until the last fragment has been received that has a zero MF bit. If one allocates a datagram buffer of maximum size, however, it should be possible to do assemble the packets as they are received. To make it more interesting, the order of fragment arrival is not guaranteed, although this is not expected to be a problem within Fermilab.

If the maximum size buffer allocation is not used, then one can build a linked list for each datagram whose fragments are in the process of being collected. Frag ments belong to the same datagram if they have the same source IP address and the same identification word in the fragment's IP header. The linked list can be maintained in order of fragment offset to simplify detection of the point at which all frag ments have been received. When the datagram is complete, another area of memory can be used to hold the complete datagram, as higher level process ing must see the entire datagram in contiguous memory.

In the local station implementation, the SNAP Task handles IP received packet processing. If the routine that processes a fragment detects a completed data gram, it assembles it and sends a reference message about it to the SNAP message queue, just as the token receive interrupt routine normally does when it receives a SNAP frame. (The SNAP protocol SAP# $AA is used currently only for IP and ARP packets on token ring.) Then SNAP will notice a larger-than-normal datagram that is unfragmented and process it normally. For either ICMP or UDP datagrams, the checksum will be done on the entire datagram.

Fragmentation

Although gateways will fragment datagrams as needed, it is considered kind for hosts to fragment outgoing datagrams so that gateways can pass the fragments without having to further trim their sizes. Fragmentation support is required to support datagram sizes that exceed the local network frame size

datagram is prepared to be sent, the transmit logic decides, based upon the destination IP address, that an MTU should be applied that is smaller than the datagram size. In this case, separate frames are built for each fragment, each one with an IP header. Each is queued in turn to the network hardware.

When fragmentation is applied, the message part of the fragment—except for the final one—must be a multiple of 8 bytes in size. (This is required because the fragment offset field in the IP header is expressed in 8-byte units.) Since the IP header part of the fragment is typically 20 bytes long, the maximum fragment size may be slightly smaller than the MTU. (This insures that the implementer is paying attention.)

The NetXmit routine assembles queued network messages into frames for delivery to the network hardware. When it has prepared the frame completely, it can make the decision on fragmentation. If the network id in the destination IP address is the local network, a different MTU is used from that used when the network is foreign. Use of different MTU values for both the local subnet and the rest of the Fermilab network is also supported.

Since the complete datagram is already assembled in the frame buffer ready to be transmitted before it is determined that it must be fragmented, the first fragment can be sent from the first part of the datagram, after adjusting for the fragment size in the IP header, setting the MF bit, and recalculating the IP checksum. The additional fragments can be built into new frame buffer space by copying the frame/IP/SNAP headers and the next fragment-size part of the message.